CLAIMS:

1. A method of managing processor tasks in a multi-processor computing system, comprising:

storing the processor tasks in a shared memory that may be accessed by a plurality of processing units of the multi-processor computing system; and

permitting the processing units to determine which of the processor tasks should be executed based on priorities of the processor tasks.

2. The method of claim 1, wherein the processor tasks that are executed are copied from the shared memory.

3. The method of claim 1, wherein the processing units comprise a main processing unit and a plurality of sub-processing units and the sub-processing units access the processor tasks in the shared memory.

4. A method of managing processor tasks in a multi-processor computing system, comprising:

storing the processor tasks in a shared memory that may be accessed by a plurality of processing units of the multi-processor computing system;

storing a task table in the shared memory, the task table including a task table entry associated with each of the processor tasks;

linking at least some of the task table entries together to achieve at least one list of processor tasks to be invoked in hierarchical order; and

permitting the processing units to use the task table to determine which of the processor tasks should be executed in accordance with the list of processor tasks.

5. The method of claim 4, wherein the processor tasks that are executed are copied from the shared memory.

6. The method of claim 4, wherein each of the task table entries includes at least one of: (i) an indication as

to whether the associated processor task is ready to be executed by one or more of the processing units; (ii) an indication as to a priority level of the associated processor task; (iii) a pointer to a previous task table entry in the list of task table entries (a previous pointer); and (iv) a pointer to a next task table entry in the list of task table entries.

7. The method of claim 4, further comprising:

storing a task queue in the shared memory, the task queue including at least one of a head pointer and a tail pointer, the head pointer providing an indication of a first one of the processor tasks in the list, and the tail pointer providing an indication of a last one of the processor tasks in the list; and

permitting the processing units to use the task table and the task queue to determine which of the processor tasks should be executed in accordance with the list of processor tasks.

8. The method of claim 7, wherein the processor tasks that are executed are copied from the shared memory.

9. The method of claim 7, further comprising:

linking respective groups of the task table entries together to produce respective lists of processor tasks, each list being in hierarchical order; and

providing that the task queue includes respective task queue entries, each entry including at least one of a head pointer and a tail pointer for each of the lists of processor tasks.

10. The method of claim 9, wherein:

each of the respective lists are associated with processor tasks of a common priority level; and

the task queue includes a task queue entry for each of a plurality of priority levels of the processor tasks.

11. The method of claim 7, wherein the plurality of processing units includes a plurality of sub-processing units, each of the sub-processing units having local memory, further comprising:

copying the task queue and the task table from the shared memory into the local memory of a given one of the sub-processing units;

searching the task queue for the head pointer to a given one of the processor tasks that is ready to be invoked; and

copying the given processor task from the shared memory to the local memory of the given sub-processing unit for execution.

12. The method of claim 11, wherein the step of searching the task queue includes searching for the head pointer to a highest priority level one of the processor tasks that is ready to be invoked.

13. The method of claim 11, further comprising: removing the given processor task from the list.

14. The method of claim 13, wherein:

each of the task table entries includes a pointer to a next task table entry; and

the removal step includes using the next pointer of the given task table entry to change the head pointer to identify the new first processor task as being ready to be next invoked.

15. The method of claim 13, wherein:

each of the task table entries includes a pointer to a previous task table entry; and

the method further includes modifying the previous pointer of the last task table entry of the list to point to the task table entry associated with the new first processor task of the list.

16. The method of claim 11, wherein:

each of the task table entries includes an indication as to whether the associated processor task is READY to be executed or is RUNNING on one or more of the sub-processing units; and

the method further includes modifying the given task table entry to indicate that the given processor task is RUNNING.

17. The method of claim 11, further comprising copying the task queue and the task table from the local memory of the given sub-processing unit into the shared memory when the given sub-processing unit has completed its use thereof.

18. The method of claim 17, further comprising: permitting the task queue and the task table to be copied from the shared memory when the given sub-processing unit has completed its use thereof.

19. A method of managing processor tasks in a multi-processor computing system, comprising:

storing the processor tasks in a shared memory that may be accessed by a plurality of processing units of the multi-processor computing system;

storing a task table in the shared memory, the task table including a task table entry associated with each of the processor tasks;

linking at least some of the task table entries together to achieve at least one list of processor tasks in hierarchical order;

at least initiating execution a first one of the processor tasks of the list within a given one of the processing units, wherein the first one of the processor tasks yields the given processing unit such that it is capable of executing another of the processor tasks; and

determining which of the other processor tasks should be executed next within the given processing unit by permitting

the given processing unit to use the task table to make such determination.

20. The method of claim 19, further comprising:

storing a task queue in the shared memory, the task queue including at least one of a head pointer and a tail pointer, the head pointer providing an indication of a new first one of the processor tasks in the list, and the tail pointer providing an indication of a last one of the processor tasks in the list; and

permitting the processing units to use the task table and the task queue to determine which of the processor tasks should be executed next.

21. The method of claim 20, wherein the plurality of processing units include a plurality of sub-processing units, each of the sub-processing units having local memory, and wherein the step of determining includes:

copying the task queue and the task table from the shared memory into a local memory of the given sub-processing unit; and

searching the task queue for the head pointer to the new first processor task that is ready to be invoked.

22. The method of claim 20, further comprising: adding the first processor task back into the list.

23. The method of claim 22, wherein:

each of the task table entries includes a pointer to a next task table entry and pointer to a previous task table entry; and

the step of adding includes modifying the linking of the task table entries to include links to the task table entry associated with the first processor task.

24. The method of claim 23, wherein the step of modifying the linking of the task table entries includes linking the task table entry associated with the first processor task between a prior task table entry and a

following task table entry that were previously linked to one another.

25. The method of claim 24, further comprising:

modifying the next pointer of the prior task table entry to point to the task table associated with the first processor task;

modifying the previous pointer of the task table entry associated with the first processor task to point to the prior task table entry;

modifying the next pointer of the task table entry associated with the first processor task to point to the following task table entry; and

modifying the previous pointer of the following task table entry to point to the task table entry associated with the first processor task.

26. A multi-processor apparatus, comprising:

a plurality of processing units, each processing unit including a local memory in which to execute processor tasks; and

a shared memory operable to store processor tasks that are ready to be executed,

wherein the processing units are operable to determine which of the processor tasks should be executed based on priorities of the processor tasks.

27. The multi-processor apparatus of claim 26, wherein the processor tasks that are executed are copied from the shared memory.

28. The apparatus of claim 26, wherein the plurality of processing units comprise a main processing unit and a plurality of sub-processing units and the sub-processor units access the processor tasks in the shared memory.

29. A multi-processor apparatus, comprising:

a plurality of processing units, each processing unit including a local memory in which to execute processor tasks; and

a shared memory operable to store: (i) processor tasks that are ready to be executed, and (ii) a task table including a task table entry associated with each of the processor tasks,

wherein the processing units are operable to use the task table to determine which of the processor tasks should be copied from the shared memory into their local memories and executed.

30. The apparatus of claim 29, wherein:

at least some of the task table entries are linked together to achieve at least one list of processor tasks to be invoked in hierarchical order; and

the processing units are operable to use the linked list to determine which of the processor tasks should be copied from the shared memory and executed.

31. The apparatus of claim 30, wherein each of the task table entries includes at least one of: (i) an indication as to whether the associated processor task is ready to be executed by one or more of the sub-processing units; (ii) an indication as to a priority level of the associated processor task; (iii) a pointer to a previous task table entry in the list of task table entries; and (iv) a pointer to a next task table entry in the list of task table entries.

32. The apparatus of claim 30, wherein:

the shared memory is further operable to store a task queue having at least one of a head pointer and a tail pointer, the head pointer providing an indication of a first one of the processor tasks in the list, and the tail pointer providing an indication of a last one of the processor tasks in the list; and

the processing units are operable to use the task table and the task queue to determine which of the processor tasks should be executed in accordance with the list of processor tasks.

33. The apparatus of claim 32, wherein the processor tasks that are executed are copied from shared memory.

34. The apparatus of claim 32, wherein:

respective groups of the task table entries are linked together to produce respective lists of processor tasks, each list being in hierarchical order; and

the task queue includes respective task queue entries, each entry including at least one of a head pointer and a tail pointer for each of the lists of processor tasks.

35. The apparatus of claim 34, wherein:

each of the respective lists are associated with processor tasks of a common priority level; and

the task queue includes a task queue entry for each of a plurality of priority levels of the processor tasks.

36. The apparatus of claim 32, wherein the plurality of processing units includes a plurality of sub-processing units, each of the sub-processing units having local memory, the sub-processing units are operable to:

copy the task queue and the task table from the shared memory into their respective local memories;

search the task queue for the head pointer to a given one of the processor tasks that is ready to be invoked; and

copy the given processor task from the shared memory to the local memory thereof for execution.

37. The apparatus of claim 36, wherein the sub-processing units are operable to search the task queue for the head pointer to a highest priority level one of the processor tasks that is ready to be invoked.

38. The apparatus of claim 36, wherein the sub-processing units are operable to remove the given processor task from the list.

39. The apparatus of claim 38, wherein:

each of the task table entries includes a pointer to a next task table entry; and

the sub-processing units are operable to modify the head pointer to identify the new first processor task as being ready to be next invoked using the next pointer of the given task table entry.

40. The apparatus of claim 38, wherein:

each of the task table entries includes a pointer to a previous task table entry; and

the sub-processing units are operable to modify the previous pointer of the last task table entry of the list to point to the task table entry associated with the new first processor task of the list.

41. The apparatus of claim 36, wherein:

each of the task table entries includes an indication as to whether the associated processor task is READY to be executed or is RUNNING on one or more of the sub-processing units; and

the sub-processing units are operable to modify the given task table entry to indicate that the given processor task is RUNNING.

42. The apparatus of claim 36, wherein the sub-processing units are operable to copy the task queue and the task table from the local memory thereof into the shared memory when the given sub-processing unit has completed its use thereof.

43. A multi-processor apparatus, comprising:

a plurality of sub-processing units, each sub-processing unit including a local memory in which to execute processor tasks; and

a shared memory operable to store: (i) processor tasks that are ready to be executed, and (ii) a task table including a task table entry associated with each of the processor tasks, wherein:

the sub-processing units are operable to at least initiate execution a first processor task and yield such that it is capable of executing another of the processor tasks, and

the sub-processing units are operable to determine which of the other processor tasks should be executed next based on the task table.

44. The apparatus of claim 43, wherein at least some of the task table entries are linked together to achieve at least one list of processor tasks in hierarchical order;

45. The apparatus of claim 44, wherein:

the shared memory is operable to store a task queue having at least one of a head pointer and a tail pointer, the head pointer providing an indication of a new first one of the processor tasks in the list, and the tail pointer providing an indication of a last one of the processor tasks in the list; and

the sub-processing units are operable to use the task table and the task queue to determine which of the processor tasks should be executed next.

46. The apparatus of claim 45, wherein the sub-processing units are operable to:

copy the task queue and the task table from the shared memory into the local memory thereof; and

search the task queue for the head pointer to the new first processor task that is ready to be invoked.

47. The apparatus of claim 45, wherein the sub-processing units are operable to add the first processor task back into the list.

48. The apparatus of claim 47, wherein:

each of the task table entries includes a pointer to a next task table entry and pointer to a previous task table entry; and

the sub-processing units are operable to modify the linking of the task table entries to include links to the task table entry associated with the first processor task.

49. The apparatus of claim 48, wherein the sub-processing units are operable to modify the linking of the task table entries by linking the task table entry associated with the first processor task between a prior task table entry and a following task table entry that were previously linked to one another.

50. The apparatus of claim 49, wherein the sub-processing units are operable to:

modify the next pointer of the prior task table entry to point to the task table associated with the first processor task;

modify the previous pointer of the task table entry associated with the first processor task to point to the prior task table entry;

modify the next pointer of the task table entry associated with the first processor task to point to the following task table entry; and

modify the previous pointer of the following task table entry to point to the task table entry associated with the first processor task.